Ruby master - Feature #12719

`Struct#merge` for partial updates

09/02/2016 09:27 AM - halogenandtoast (Matthew Mongeau)

Status:	Feedback
Priority:	Normal
Assignee:	
Target version:	

Description

Other languages have operators for performing partial updates on maps. I feel like Struct could be more useful if it provided an easy way of performing partial (or full) updates.

After the change you can do the following:

```
Point = Struct.new(:x, :y)

p = Point.new(1, 2)
p2 = p.merge(y: 4)
p3 = p2.merge(x: 10)

p.inspect     # => #<struct Point x=1, y=2>
p2.inspect    # => #<struct Point x=1, y=4>
p3.inspect    # => #<struct Point x=10, y=4>

p.merge!("x" => 9)

p.inspect     # => #<struct Point x=9, y=2>
```

History

#1 - 09/02/2016 09:28 AM - halogenandtoast (Matthew Mongeau)

- Tracker changed from Bug to Feature

#2 - 09/02/2016 09:32 AM - halogenandtoast (Matthew Mongeau)

If this is well received I think a similar syntax could be used for hashes in place of merge.

#3 - 09/02/2016 01:11 PM - halogenandtoast (Matthew Mongeau)

- File struct_merge.patch added

As an alternative since the | syntax might get shot down. Here's a patch adding a merge function instead:

```
Point = Struct.new(:x, :y)

p = Point.new(1, 2)
p2 = p.merge(y: 4)
p3 = p2.merge(x: 10)

puts p.inspect # => #<struct Point x=1, y=2>
puts p2.inspect # => #<struct Point x=1, y=4>
puts p3.inspect # => #<struct Point x=10, y=4>
```

#4 - 09/02/2016 03:38 PM - nobu (Nobuyoshi Nakada)

In your example, the value in the LHS is ignored when the same key is present in the RHS hash.
It doesn't feel nice as | operator.

merge sounds nice in that sense, but your patch would segfault at p.merge(0).

#5 - 09/02/2016 04:22 PM - halogenandtoast (Matthew Mongeau)

Thanks, nice catch. I'll update this tomorrow to not segfault.

#6 - 09/03/2016 01:37 AM - halogenandtoast (Matthew Mongeau)

- File struct_merge_no_segfault.patch added

Updated so it won't segfault

#7 - 09/05/2016 01:22 AM - halogenandtoast (Matthew Mongeau)

- File merge_bang.patch added

Since merge closely resembles the similar hash function, I think it makes sense to also add merge! as a function. I'm not a fan of the mutating methods, but I would find it surprising if this interface was different. Here's an updated patch. I also fixed some of the documentation I wrote.

#8 - 09/05/2016 02:53 AM - halogenandtoast (Matthew Mongeau)

- File deleted (struct_update.patch)

#9 - 09/05/2016 02:53 AM - halogenandtoast (Matthew Mongeau)

- File deleted (struct_merge.patch)

#10 - 09/05/2016 02:53 AM - halogenandtoast (Matthew Mongeau)

- File deleted (struct_merge_no_segfault.patch)

#11 - 09/05/2016 02:58 AM - halogenandtoast (Matthew Mongeau)

- Description updated

- Subject changed from `Struct#|` for partial updates to `Struct#merge` for partial updates

- File struct_merge.patch added

Update ChangeLog

#12 - 09/05/2016 02:58 AM - halogenandtoast (Matthew Mongeau)

- File deleted (merge_bang.patch)

#13 - 11/25/2016 08:59 AM - matz (Yukihiro Matsumoto)

- Status changed from Open to Feedback

I want to see a real-world use-case for the feature.
In addition, I don't think the name merge is the best for the functionality.

Matz.

#14 - 12/15/2016 10:41 AM - Eregon (Benoit Daloze)

Scala has "copy" for this purpose: some_case_class_object.copy(field: new_value)

Files

struct_merge.patch	2.93 KB	09/05/2016	halogenandtoast (Matthew Mongeau)